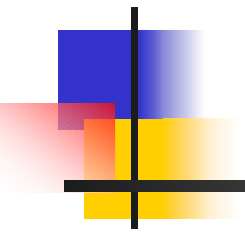
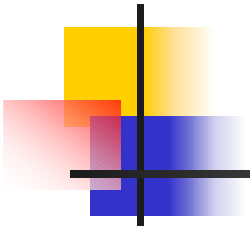


FBSNG Overview

Jim Fromm, Krzysztof Genser, Tanya Levshina, Igor Mandrichenko

Farms and Clustered Systems Group,
Computing Division,
Fermilab



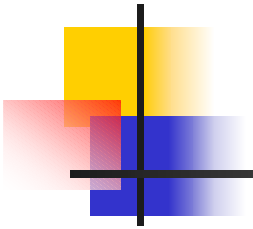
Acknowledgements

FBS developers:

- Mark Breitung
- Marilyn Schweitzer

FBS users and FBSNG testers:

- Antonio Wong Chan (academia Sinica, Taiwan, CDF collaboration)
- Yen-Chu Chen (academia Sinica, Taiwan, CDF collaboration)
- Thomas Las (Manooka, IL Junior High School)
- Miroslav Siket (academia Sinica, Taiwan, CDF collaboration)
- Heidi Schellman (northwestern university, Illinois, D0 collaboration)
- Steve Wolbers (Fermilab, CDF collaboration)
- Ping Yeh (academia Sinica, Taiwan, CDF collaboration)



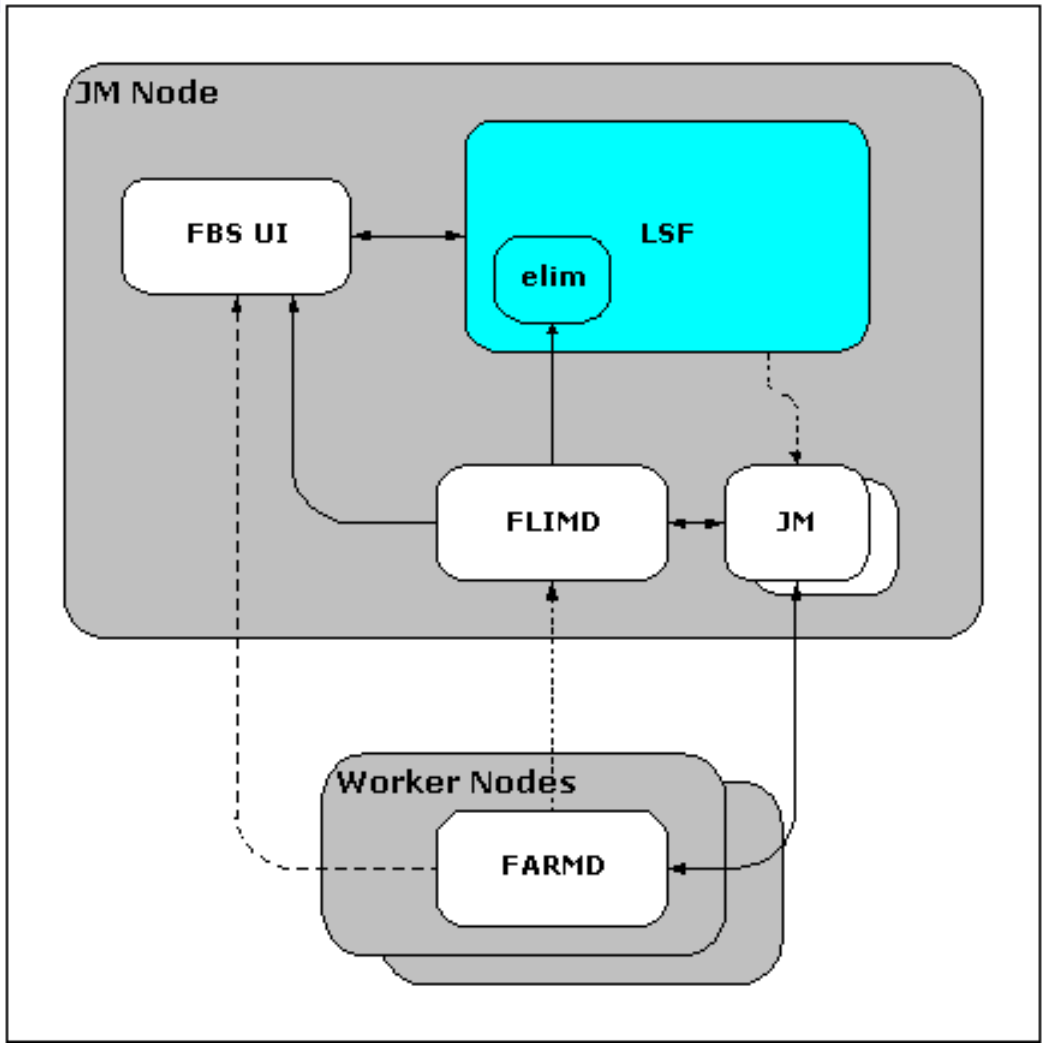
History and Status

- FBS project goals
  - Replace CPS batch in PC farms environment
  - Develop a farm batch system for file-based parallel data processing style adopted for RunII
  - Do not preclude even-based parallelism
- Milestones
  - Spring 1998 - initial FBS design, first working prototypes
  - Fall 1998 - first production users (E871)
  - Spring 1999 - review by CDF, D0
  - Fall 1999 - FBS v2.2
  - Fall 1999 - beginning of FBSNG project
  - July 2000 - FBSNG v1.0 released
- Currently FBSNG is installed on:
  - Fixed target farm fnsfo
  - CDF farm
  - NIKHEF (D0 collaborators)

FBSNG Project Goals

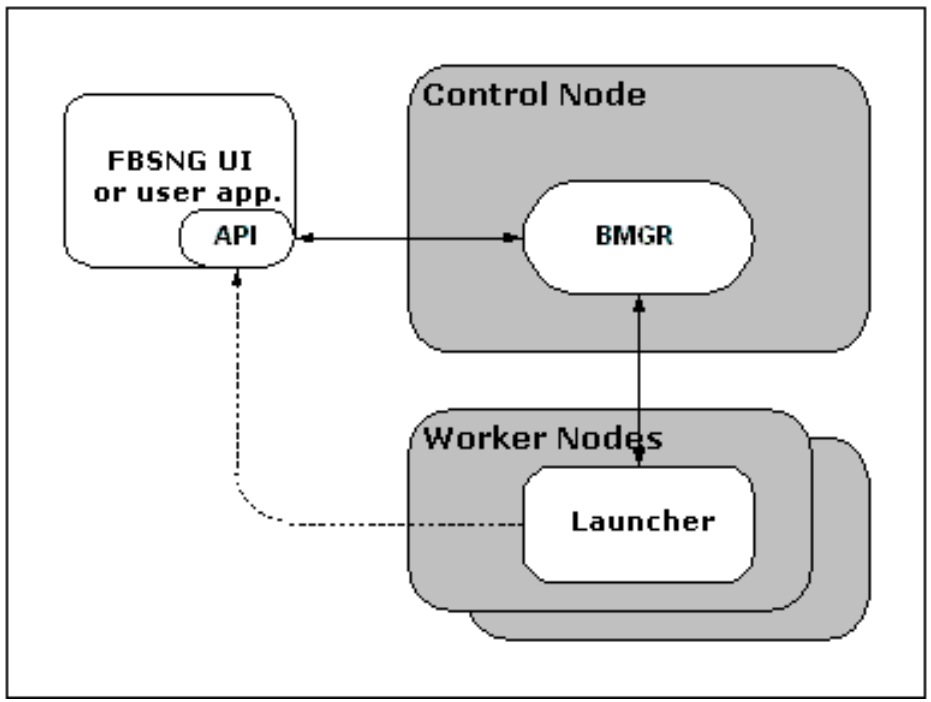
- Immediate
  - Stop using LSF as scheduler and job storage
    - Reduce maintenance and support costs
    - Avoid possible scalability problems
  - Release first version of FBSNG early
    - "Do not brake anything!"
    - Preserve as many features as reasonably possible
    - Add few fundamental features
      - Abstract resources
      - Customizable scheduler
    - Basis for evaluation and future development
- Long-term
  - Having FBSNG as integral product creates possibilities for future development
    - Dynamic re-configuration
    - Further development of resource management
    - Integration with FIPC
    - Open system (API)

FBS Design (Big Picture)



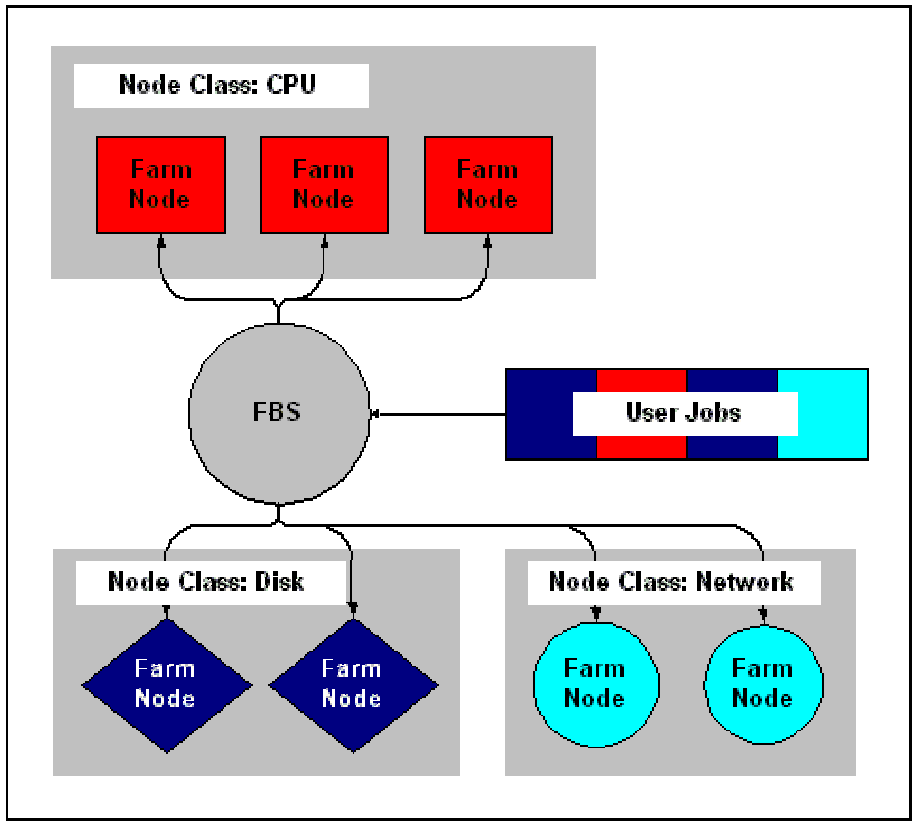
- LSF is used for
  - Scheduling
  - Job storage
  - Some resource management

FBSNG Design (Big Picture)

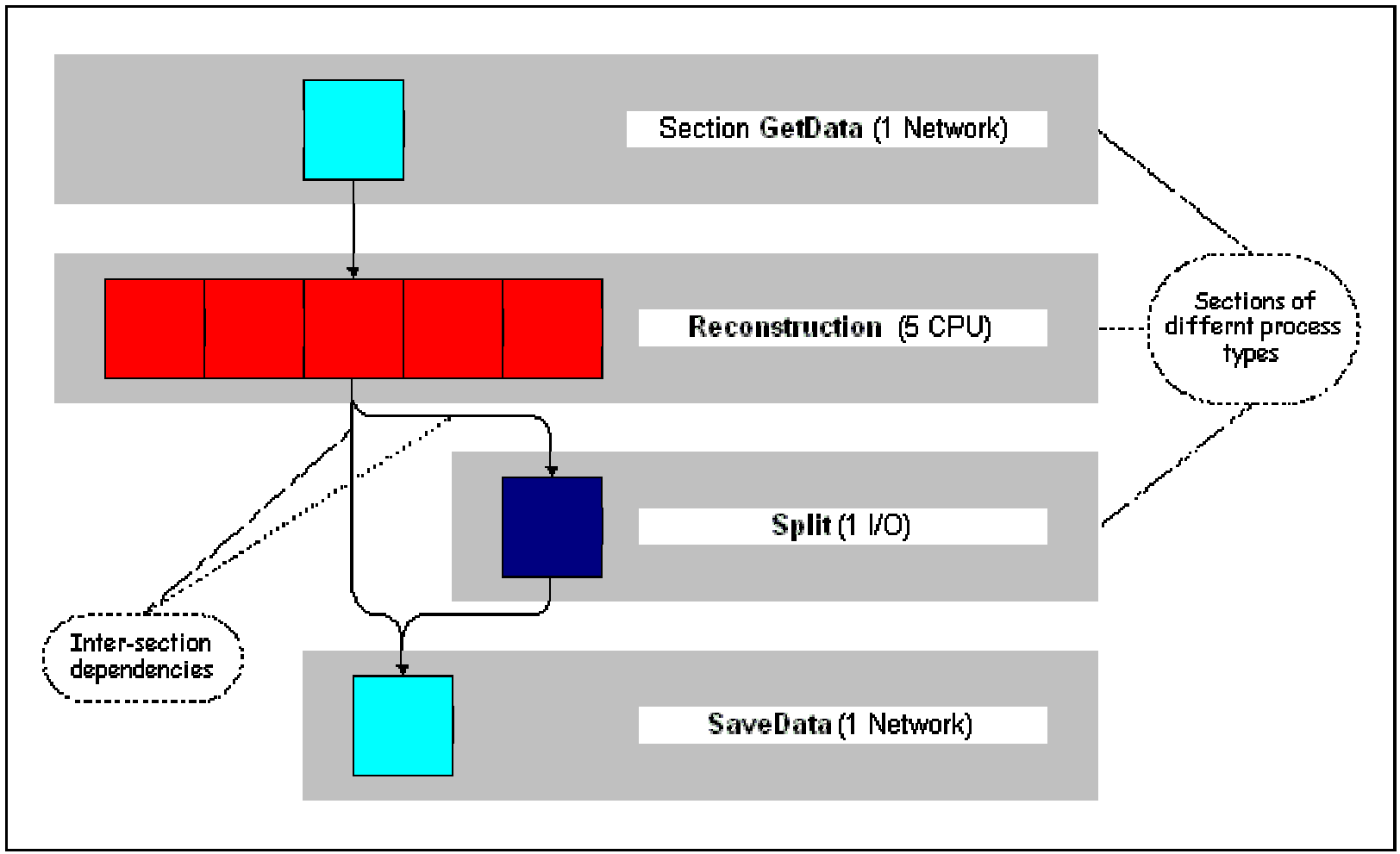


- BMGR functions:
  - Scheduling
  - Resource management
  - Job storage
  - Communication with API clients

FBS Concepts: Farm Model

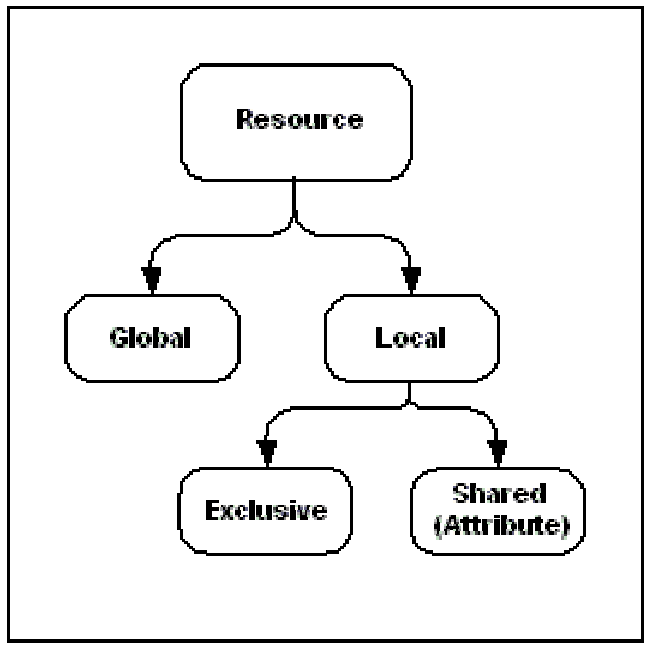


FBS Concepts: Job Consists of Sections

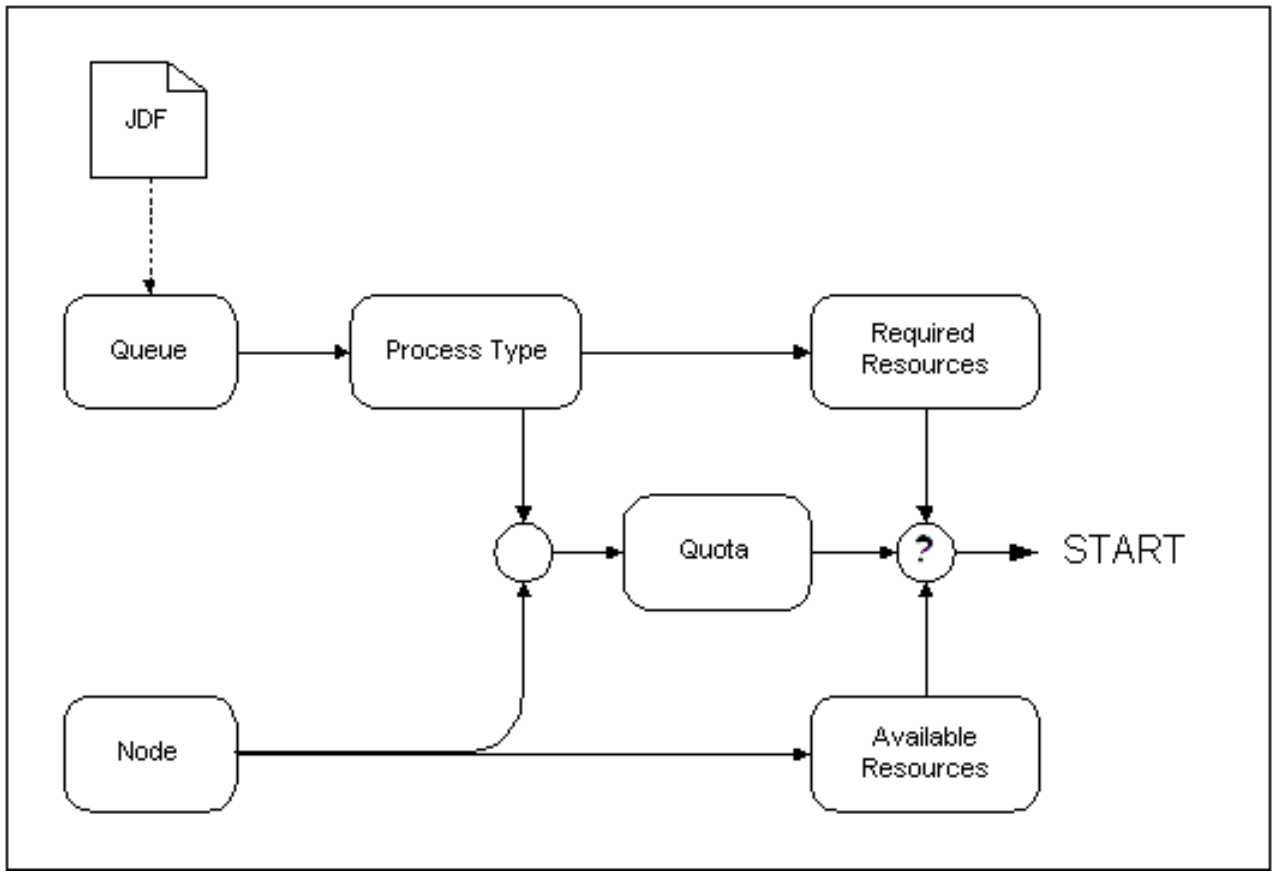


What is new: Resources

- Global resources
  - Hosted by whole farm
  - Available on any node
  - Examples:
    - Disk space on NFS server
    - Network bandwidth
- Local resources
  - Hosted by individual nodes
  - Available only on the node
  - Examples:
    - CPU
    - Local disk
    - Tape drives
- Node attributes
  - Unlimited local resources
  - Examples:
    - Special software installed
    - Version of OS
- See also http://www-isd.fnal.gov/fbs/FBS2/FBSNG_Resources.htm

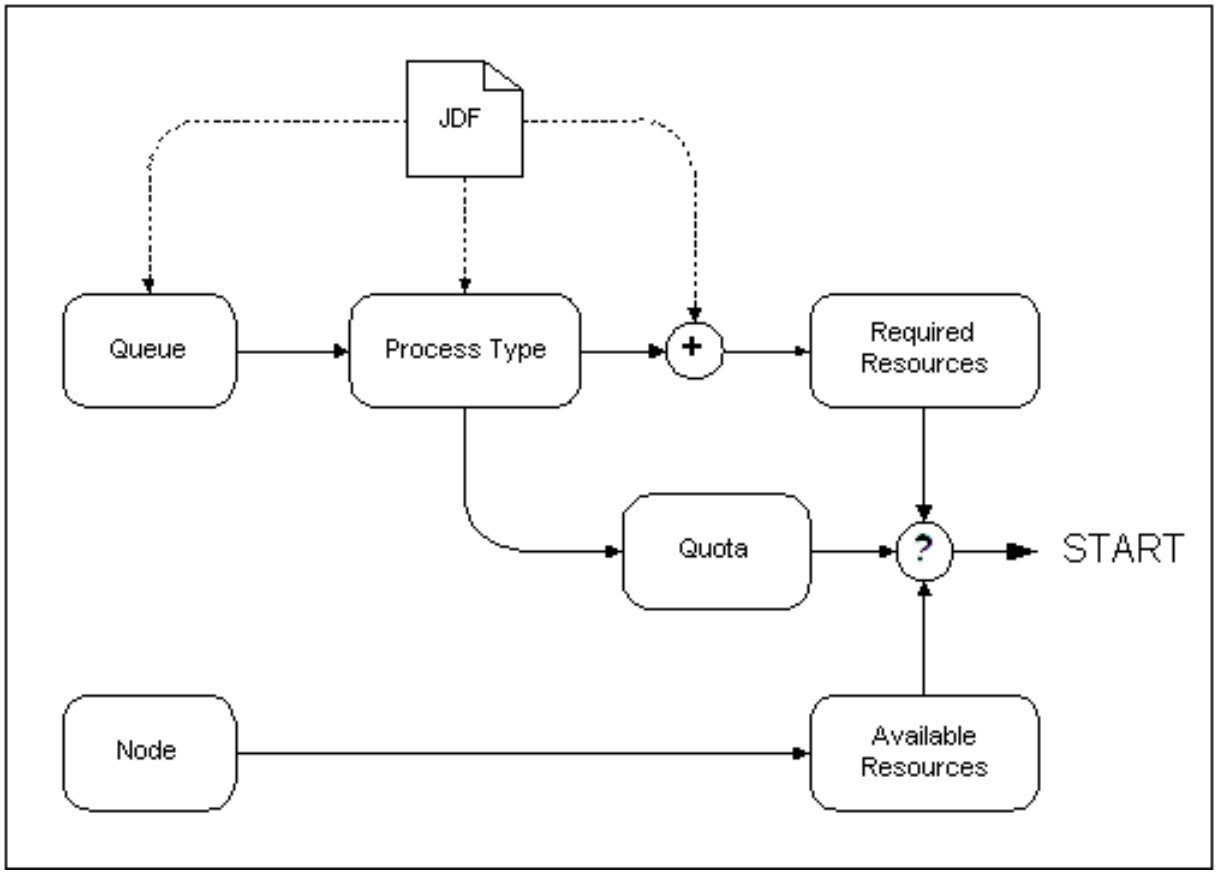


FBS Resources Management

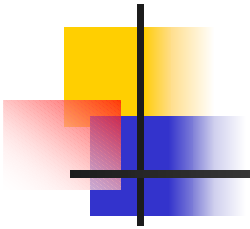


- Process type per task or project
- Queue statically associated with process type and resources
- *Therefore,* queue per task or project
- Per-node per-process type quotas

FBSNG Resource Management



- Process type per task or project
- Soft association between queue and process type (user can override)
- User can request additional resources
- Queue is more of a scheduling than resource management entity
- Generic queues
- More flexibility for users
- Only per-process type resource quotas



FBSNG Users Interface

- Job control
  - Submit – *jobs can be submitted from any node of the farm*
  - Monitor status
  - Kill/cancel
  - Hold/release
  - History
  - E-mail notification
- Farm node status
- *Resource utilization statistics*
- *Scheduler status*

FBSNG: Example of Job Description File (JDF)

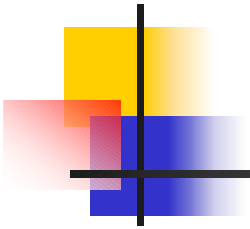
```
SECTION Stage                          First section: pre-stage data
QUEUE=IO_Q                             Queue to submit to
EXEC=stage.sh VSN123 /stage01          Command to execute
NUMPROC=1                              1 process

SECTION Reconstructor                  Second section: reconstruction
QUEUE=Long_Q
EXEC=reco123.sh /stage01 /stage02
NUMPROC=10                             10 processes
DEPEND=done(Stage)                     Only if pre-staging succeeds
PROC_RESOURCES=disk:5 Linux Worker     5GB of local disk, Linux, Worker node

SECTION Clean-up                       Emergency clean-up section
QUEUE=Short_Q
PROC_TYPE = Light                      Override default process type
EXEC=clean.sh /stage02 VSN123
NUMPROC=1
PRIO_INC = 5                           Run at higher priority
DEPEND=failed(Reconstructor)           Run only if reconstructor fails
```

FBSNG Batch Process Environment

- Environment Variables
  - FBS_JOB_ID
  - FBS_SECTION_NAME
  - FBS_JOB_SIZE - number of processes in the section
  - FBS_PROC_NO - logical process id (1…FBS_JOB_SIZE)
  - FBS_SCRATCH - scratch directory created for the process
  - HOME - home directory
- Current working directory is HOME
- Stderr, stdout as specified in JDF
- Unique UNIX session ID



User Interface: What is old ?

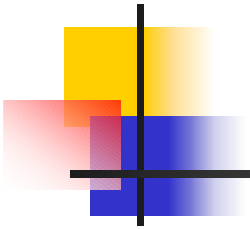
- Obsolete:
  - JDF END_TIME field
  - “move” command (replaced with “cprio”)

User Interface: What has changed ?

- “farms” -> “fbs”, “fbsng” - synonyms
  - e.g. “farms submit” -> “fbs submit”
- Jobs can be submitted from any farm node
  - Submit a job from another job
- JDF:
  - BEGIN_TIME -> HOLD_TIME, time specification format
    - Absolute: hold till noon tomorrow
    - Relative: hold for 10 minutes
  - JOB_STDOUT -> SECT_STDOUT
    - Unprotected directory required
    - “slog” command
  - LOCAL_DISK -> PROC_RESOURCES
    - e.g. “LOCAL_DISK=5” -> “PROC_RESOURCES = disk:5 tape:1”
- Section ID
  - Main.1234 -> 1234.Main

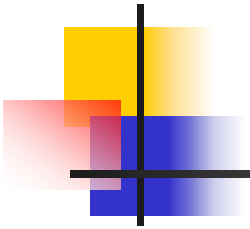
User Interface: What is new ?

- JDF:
  - PROC_TYPE – override Queue default
  - PROC_RESOURCES – resources per process
  - SECT_RESOURCES – resources per section
  - PRIO_INC – submit the section at higher priority
- New commands:
  - ls, lj – list jobs, sections (scheduling parameters)
  - cprio – change section priority
  - slog – show section log
  - hold/release job/section/queue
  - ptypes, resources – statistics on process types, resources
- See v1.0 Release Notes
  - http://www-isd.fnal.gov/fbs/FBS2/FBSNG_RelNotes.htm



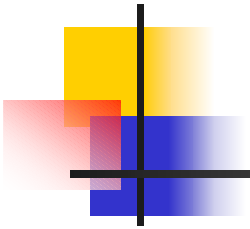
FBSNG Scheduler

- Algorithms are based on the idea of dynamic priorities
- Controllable fair-share scheduling
  - Projects are assigned relative shares of farm resources
- Guaranteed scheduling
  - No infinite delays for *big* jobs
  - Will hold *small* jobs if necessary
- Customizable
  - Wide variety of configuration options
- See also http://www-isd.fnal.gov/fbs/FBS2/FBSNG_Scheduler.htm



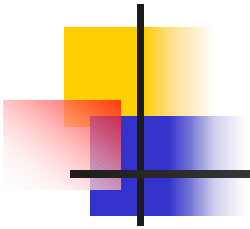
FBSNG API

- Job submission
  - FBSClient
  - FBSJobDesc
  - FBSSectionDesc
- Job monitoring and control
  - FBSJobInfo
  - FBSSectionInfo
  - FBSProcessInfo
- Resource management and monitoring
  - FBSQueueInfo
  - FBSNodeInfo
  - FBSNodeClassInfo
  - FBSProcessTypeInfo
- Python binding available
- UI, GUI are implemented in terms of API



FBSNG Requirements

- LSF no longer required
- On control node:
  - Bmgr daemon (non-root)
  - Logd daemon (optional, non-root)
- On each worker node:
  - Launcher (root)
  - Rstatd (optional)
- Software/hardware requirements:
  - Pyhton (99% FBSNG sources are python)
  - Tcl/Tk, Tkinter (for GUI)
  - FCSLIB (common Python module library)
  - Configuration file synchronized on all worker nodes (NFS works well)



FBSNG Project Status and Plans for Future

- FBSNG V1.0 (IRIX, Linux) released July 6th
- Soon to be available from Fermitools
- V1.0 is evaluation version
  - Solicit feedback (suggestions, criticism) from users
    - http://www-isd.fnal.gov/fbs/FBS2/fut_feat.html
  - Make plans based on such feedback